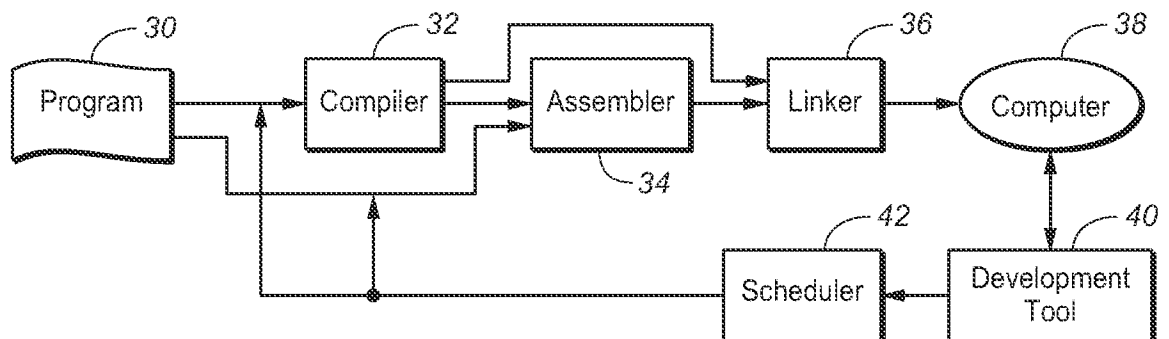
**FIG. 1****FIG. 2**

REPLACEMENT SHEET

2 / 5

*bp = (*bp & rm[offset]) | ((cd<<offset) & mask) ← 50

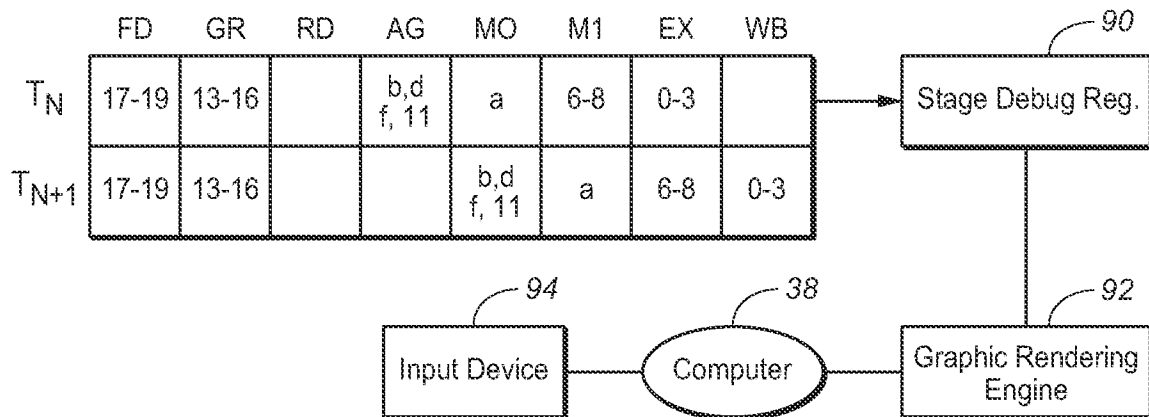
1:	const	tmp1,mask	;load mask	} 52
2:	consth	tmp1,mask		
3:	sll	tmp2,cd,offset	;cd<<offset	
4:	and	tmp2,tmp1,tmp2	;(cd<<offset) & mask	
5:	const	tmp1,_rm	;base address rm	
6:	consth	tmp1,_rm		
7:	add	tmp1,tmp1,offset	;address of rm[offset]	
8:	load	tmp1,(tmp1)	;rm[offset]	
9:	load	bp,(&bp)	;*bp	
10:	and	tmp1,bp,tmp1	;*bp & rm[offset]	
11:	or	tmp1,tmp1,tmp2	;final expression	
12:	store	tmp1,(&bp)	;assign *bp	

Decode				Execute					Writeback		Cycle
I0	I1	I2	I3	ALU1	ALU2	SHF	LS	BRN	R1	R2	
1:const	2:consth	3:sll	4:and								1
5:const	6:consth	7:add	8:load	1:const		3:sll					2
9:load	10:and	11:or	12:store		2:consth				1	3	3
				4:and	5:const				2		4
				6:consth					4	5	5
					7:add				6		6
							8:load		7		7
							9:load				8
									8		9
	10:and								9		10
		11:or							10		11
			12:store						11		12

FIG. 3

Stage	Issue 1	Issue 2	Issue 3	Issue 4
Write Back	None	None	None	None
Execute	0x1000	0x1001	0x1002	0x1003
Memory 1	0x1006	0x1007	0x1008	None
Memory 0	0x100A	None	None	None
Address Generation	0x100B	0x100D	0x100F	0x1011
Read Data	None	None	None	None
Grouping	0x1013	0x1014	0x1015	0x1016
Fetch/Decode	0x1017	0x1018	0x1019	None

60

FIG. 4**FIG. 7**

REPLACEMENT SHEET

4 / 5

11 cycles		TARGET DISASSEMBLED CODE	
--	0xffff	:	nop
--	0x1000	EX	_start:: bits %fmode, 0x2
--	0x1001	EX	: mov r9, 2
--	0x1002	EX	: mov r10, 0xffff00
--	0x1003	EX	: mov r11, 0xf0f0
--	0x1006	M1	: mov r2, r9
--	0x1007	M1	: mov r3, r9
--	0x1008	M1	: mov r13, 0x14
--	0x100a	M0	tst_bgn: : xor.e r0, r0
--	0x100b	AG	: mov g0, r0
--	0x100d	AG	: mov %loop0, 0x1c
--	0x100f	AG	: mov r8, 0x3d
--	0x1011	AG	: mov r12, 0x0
--	0x1013	GR	lp_calc: : ldu r4, r12, +1
--	0x1014	GR	: ldu r6, r8, +1
--	0x1015	GR	: mac r0, r4, r6
--	0x1016	GR	: ldu r5, r12, +1
--	0x1017	FD	: ldu r7, r8, +1
--	0x1018	FD	: mac r0, r5, r7

FIG. 5

Disassembly			TARGET DISASSEMBLED CODE
469 cycles			
-- 0x1e0		:	std r4, a7, 0x5
-- 0x1e1	GR	:	ldd r2, a7, 0x1
-- 0x1e2	GR	:	mov a6, a7
-- 0x1e3	GR	:	add a6, 3
-- 0x1e4	FD	:	ldd r4, a6
-- 0x1e5	WBFD	:	mov.e a1, r2
-- 0x1e6	M1FD	:	mov.e a0, r4
-- 0x1e7	MOFD	:	ld r6, a0
-- 0x1e8	M0	:	st r6, a1
-- 0x1e9	AG	:	ld r6, a1
-- 0x1ea	AG	:	iadd.e r4, 1
-- 0x1ec	AG	:	std r4, a6
-- 0x1ed	AG	:	iadd.e r2, 1
-- 0x1ef	RD	:	std r2, a7, 0x1
-- 0x1f0	RD	:	cmp r6, 0
-- 0x1f1	GR	:	bnz 0x1e1
-- 0x1f2		:	ldd a0, a7, 0x5
-- 0x1f3		:	add a7, 6
-- 0x1f4			__FUNC_EXIT_strcpy:: ret
-- 0x1f5			write sdsp:: pushd r10, a7

FIG. 6